日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 Date of Application:	２００３年　４月２３日
出願番号 Application Number: [ST. 10/C]:	特願２００３－１１８６０２ [ＪＰ２００３－１１８６０２]
出願人 Applicant(s):	株式会社日立製作所

U.S. Appln. Filed 4-22-04
Inventor: M. Mitsumori et al
Mathingly Stanger & Malur
Docket No. IT-421

２００４年　４月　８日

特許庁長官
Commissioner,
Japan Patent Office
今井康夫

出証番号　出証特２００４－３０２９１３６

【書類名】　特許願

【整理番号】　NT03P0209

【提出日】　平成15年 4月23日

【あて先】　特許庁長官　殿

【国際特許分類】　G06F 12/08

【発明者】

【住所又は居所】　神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内

【氏名】　三森　征人

【発明者】

【住所又は居所】　神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内

【氏名】　中島　恵

【発明者】

【住所又は居所】　神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内

【氏名】　家坂　聡

【特許出願人】

【識別番号】　000005108

【氏名又は名称】　株式会社日立製作所

【代理人】

【識別番号】　100068504

【弁理士】

【氏名又は名称】　小川　勝男

【電話番号】　03-3661-0071

【選任した代理人】

【識別番号】　100086656

【弁理士】

【氏名又は名称】　田中　恭助

【電話番号】　03-3661-0071

【選任した代理人】

【識別番号】　100094352

【弁理士】

【氏名又は名称】　佐々木　孝

【電話番号】　03-3661-0071

【手数料の表示】

【予納台帳番号】　081423

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　不正メモリアクセス検知方法及びそのプログラム

【特許請求の範囲】

【請求項１】

独自のメモリ管理機能を有する言語システムと、前記言語システムの制御下で実行され前記言語システムが確保した第１のメモリ領域にアクセスする第１のプログラムコードと、ＯＳの制御下で直接実行され前記ＯＳが確保した第２のメモリ領域にアクセスする第２のプログラムコードとが実行されるコンピュータにおいて、前記第２のプログラムコードによる前記第１のメモリ領域への不正なメモリアクセスを検知するための前記言語システムによって実行される方法であって、

前記言語システムは、前記第１のプログラムコードが前記第２のプログラムコードを呼び出す前に前記第１のメモリ領域についてメモリプロテクションを設定し、前記第２のプログラムコードを呼び出して実行させ、メモリプロテクション例外が発生したとき前記第２のプログラムコードによる不正メモリアクセスを外部に報告し、前記第２のプログラムコードの実行が終了して前記言語システムに制御が戻ったときに前記メモリプロテクションを解除することを特徴とする不正メモリアクセス検知方法。

【請求項２】

前記言語システムは、さらに前記メモリプロテクション例外が発生したとき、前記第１のプログラムコードによる前記第１のメモリ領域への正常なメモリアクセスであることを検出した場合には、前記メモリプロテクションを解除し、前記正常なメモリアクセスを許し、再び前記メモリプロテクションを設定することを特徴とする請求項１記載の不正メモリアクセス検知方法。

【請求項３】

前記第１のプログラムコードがマルチスレッド制御下で実行される場合に、前記言語システムは、あるスレッドが前記第２のプログラムコードを呼び出す間、他のスレッドの実行を停止させることを特徴とする請求項１記載の不正メモリアクセス検知方法。

【請求項４】

独自のメモリ管理機能を有する言語システムと、前記言語システムの制御下で実行され前記言語システムが確保した第１のメモリ領域にアクセスする第１のプログラムコードと、ＯＳの制御下で直接実行され前記ＯＳが確保した第２のメモリ領域にアクセスする第２のプログラムコードとが実行されるコンピュータにおいて、前記第２のプログラムコードによる前記第１のメモリ領域への不正なメモリアクセスを検知するための前記言語システムによって実行される方法であって、

前記言語システムは、前記第１のプログラムコードが前記第２のプログラムコードを呼び出す前に前記第１のメモリ領域についてその内容に対応するコード情報を保存し、前記第２のプログラムコードを呼び出して実行させ、前記第２のプログラムコードの実行が終了して前記言語システムに制御が戻ったときに前記第１のメモリ領域の内容に対応するコード情報が保存されたコード情報と一致するか否かを判定し、不一致の場合に前記第２のプログラムコードによる不正メモリアクセスを外部に報告することを特徴とする不正メモリアクセス検知方法。

【請求項５】

前記言語システムは、さらに前記第２のプログラムコードの呼び出し中に前記第１のプログラムコードによる前記第１のメモリ領域への正常なメモリ更新を検出したとき、更新された前記第１のメモリ領域の内容に対応するコード情報によって前記保存されたコード情報を更新することを特徴とする請求項４記載の不正メモリアクセス検知方法。

【請求項６】

前記第１のプログラムコードがマルチスレッド制御下で実行される場合に、前記言語システムは、あるスレッドが前記第２のプログラムコードを呼び出す間、他のスレッドの実行を停止させることを特徴とする請求項４記載の不正メモリアクセス検知方法。

【請求項７】

独自のメモリ管理機能を有する言語システムと、前記言語システムの制御下で実行され前記言語システムが確保した第１のメモリ領域にアクセスする第１のプ

ログラムコードと、ＯＳの制御下で直接実行され前記ＯＳが確保した第２のメモリ領域にアクセスする第２のプログラムコードとが実行されるコンピュータにおいて、前記コンピュータに前記第２のプログラムコードによる前記第１のメモリ領域への不正なメモリアクセスを検知する前記言語システムの機能を実現させるためのプログラムであって、

前記コンピュータに、前記第１のプログラムコードが前記第２のプログラムコードを呼び出す前に前記第１のメモリ領域についてメモリプロテクションを設定する機能、前記第２のプログラムコードを呼び出して実行させる機能、メモリプロテクション例外が発生したとき前記第２のプログラムコードによる不正メモリアクセスを外部に報告する機能、および前記第２のプログラムコードの実行が終了して前記言語システムに制御が戻ったときに前記メモリプロテクションを解除する機能を実現させるためのプログラム。

【請求項８】

前記コンピュータに、さらに前記メモリプロテクション例外が発生したとき、前記第１のプログラムコードによる前記第１のメモリ領域への正常なメモリアクセスであることを検出した場合には、前記メモリプロテクションを解除する機能、前記正常なメモリアクセスを許す機能、および再び前記メモリプロテクションを設定する機能を実現させることを特徴とする請求項７記載のプログラム。

【請求項９】

前記第１のプログラムコードがマルチスレッド制御下で実行される場合に、前記コンピュータに、あるスレッドが前記第２のプログラムコードを呼び出す間、他のスレッドの実行を停止させる機能を実現させることを特徴とする請求項７記載のプログラム。

【請求項１０】

独自のメモリ管理機能を有する言語システムと、前記言語システムの制御下で実行され前記言語システムが確保した第１のメモリ領域にアクセスする第１のプログラムコードと、ＯＳの制御下で直接実行され前記ＯＳが確保した第２のメモリ領域にアクセスする第２のプログラムコードとが実行されるコンピュータにおいて、前記コンピュータに前記第２のプログラムコードによる前記第１のメモリ

領域への不正なメモリアクセスを検知する前記言語システムの機能を実現させるためのプログラムであって、

前記コンピュータに、前記第１のプログラムコードが前記第２のプログラムコードを呼び出す前に前記第１のメモリ領域についてその内容に対応するコード情報を保存する機能、前記第２のプログラムコードを呼び出して実行させる機能、前記第２のプログラムコードの実行が終了して前記言語システムに制御が戻ったときに前記第１のメモリ領域の内容に対応するコード情報が保存されたコード情報と一致するか否かを判定する機能、および不一致の場合に前記第２のプログラムコードによる不正メモリアクセスを外部に報告する機能を実現させるためのプログラム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、不正メモリアクセス検知方法に係り、特に不正なメモリアクセスが発生しないシステムと、自由にメモリアクセスを行えるシステムとが混在する環境において好適な不正メモリアクセス検知方法に関する。

【０００２】

【従来の技術】

オブジェクト指向言語として一般的に知られているＪａｖａプログラムを実行する環境であるＪａｖａＶＭは、Ｊａｖａプログラムを実行する際に使用するメモリ領域の管理を独自に行っており、Ｊａｖａプログラムが実行されている限りにおいては不正なメモリアクセスが発生しないシステムとなっている（Ｊａｖａは、米国Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，　Ｉｎｃ．の登録商標である）。しかしＪａｖａプログラムを実行する上で必要に応じて呼び出される他言語（例えばＣ言語）で作成されたプログラムの実行中は、ＯＳがメモリ領域の管理を行っているため、その間はＪａｖａＶＭでは不正なメモリアクセスが発生したかどうかの検出ができない。そのためＪａｖａプログラムから呼び出された他言語で作成されたプログラムが誤ってＪａｖａＶＭの管理しているメモリ領域を不正にアクセスして更新してしまう可能性がある。しかしこのような不正メモリアクセ

スを早期に検出する技術は知られていない。

【０００３】

なおこの種の技術として関連するものには、例えば特開平６－４４１２９号公報（特許文献１）、特開平５－２８０５３号公報（特許文献２）などがある。

【０００４】

【特許文献１】

特開平６－４４１２９号公報

【特許文献２】

特開平５－２８０５３号公報

【０００５】

【発明が解決しようとする課題】

上記の従来技術は、Ｊａｖａプログラムを実行する上で必要に応じて呼び出される他言語で作成されたプログラムの実行時に、他言語プログラムがＪａｖａＶＭで管理しているメモリ領域を不正にアクセスして更新した場合、その後に実行されるＪａｖａプログラムがそのメモリ領域にアクセスし異常状態となるまでは不正なメモリアクセスが発生していたことを検知できず、問題のあるプログラムを特定することが困難であった。

【０００６】

本発明の目的は、不正なメモリアクセスが発生しないシステムで動作するプログラムから呼ばれた自由にメモリアクセスを行えるシステムで動作するプログラムが不正なメモリアクセスをしたことを早期に検出することにある。

【０００７】

【課題を解決するための手段】

本発明は、不正なメモリアクセスが発生しないシステムで動作しているプログラムから呼ばれた自由にメモリアクセスが行えるシステムで動作するプログラムが実行中又は実行後早期に不正なメモリアクセスが発生したことを検出する技術を特徴とする。

【０００８】

【発明の実施の形態】

不正なメモリアクセスが発生しないシステムとしてＪａｖａＶＭを例とし、そのシステムで動作するプログラムとしてＪａｖａで記述したプログラムを例とし、自由にメモリアクセスを行えるシステムで動作するプログラムとしてＣ言語で記述したネイティブメソッドライブラリを使用する場合を例とする実施の形態について以下図面を用いて説明する。

【０００９】

図１は、実施形態のＪａｖａＶＭの構成とその入力ファイルを示す図である。１０１は記憶装置上に格納されているＪａｖａソースプログラムである。１０２はＪａｖａソースプログラムをＪａｖａＶＭで実行できるように中間言語で記述されるバイトコードに変換するＪａｖａコンパイラである。１０３はＪａｖａコンパイラで作成したバイトコードが格納されているＪａｖａクラスファイルである。１０４はバイトコードを実行する際に必要な他のバイトコード、すなわちクラスファイルを収めたクラスライブラリである。１０５はバイトコードから呼び出されるＪａｖａ言語以外の言語で記述されたネイティブメソッドライブラリである。１０６は本発明を実施する言語システムであるＪａｖａＶＭの本体である。１０７はＪａｖａクラスファイル１０３のバイトコードをメモリにロードするバイトコード読み取り部である。１０８は入力されたバイトコードやネイティブメソッドライブラリ１０５を呼び出して、Ｊａｖａプログラムを実際に実行する実行部である。実行部１０８は、バイトコードを実行する際に必要な他のバイトコードを収めたクラスファイルをロードするクラスライブラリロード部１０９、Ｊａｖａ言語以外の言語で記述されたネイティブメソッドライブラリ１０５をメモリにロードするネイティブメソッドライブラリロード部１１０、Ｊａｖａプログラムを実行する際にＪａｖａＶＭが必要なメモリ領域を確保するメモリ確保部１１１、バイトコードを実行するバイトコード実行部１１２、ネイティブメソッドライブラリ１０５を実行するネイティブメソッドライブラリ実行部１１３、ネイティブメソッドライブラリの実行中、または実行後に不正なメモリアクセスが発生したかどうかを検出する不正メモリアクセス検出部１１４から構成される。

【００１０】

なお図示していないが、ＪａｖａＶＭ１０６はＯＳ（オペレーティングシステ

ム）の制御を受ける。このＯＳはネイティブのメモリ管理機能を有する。ネイティブメソッドライブラリは、このＯＳのもつメモリ管理機能を利用して自身のメモリ領域を確保する。言うまでもないが、図１のＪａｖａコンパイラ１０２、ＪａｖａＶＭ１０６及びＯＳは、ＣＰＵ、メモリ、記憶装置、入力装置、表示装置などをもつコンピュータのＣＰＵによって実行されるプログラムである。Ｊａｖａソースプログラム１０１、Ｊａｖａクラスファイル１０３、クラスライブラリ１０４及びネイティブメソッドライブラリ１０５は、この記憶装置に格納されるプログラムコードである。Ｊａｖａクラスファイル１０３、クラスライブラリ１０４及びネイティブメソッドライブラリ１０５は、このコンピュータによって実行されるプログラムコードである。

【００１１】

入力装置などからコマンドが投入されることによってＪａｖａＶＭ１０６が起動され、ＪａｖａＶＭ１０６が実行開始される。バイトコード読み取り部１０７は、Ｊａｖａクラスファイル１０３からバイトコードを読み込むと、実行部１０８に制御を移し、バイトコード実行部１１２が読み込まれたバイトコードを実行する。

【００１２】

図２は、実行部１０８のうち本発明に関係する部分の処理手順を示すフローチャートである。ステップ２０１及び２０２は、バイトコードを実行する際に必要なクラスライブラリ１０４をメモリにロードするクラスライブラリロード部１０９の処理である。ステップ２０３及び２０４は、バイトコードを実行する際に必要なネイティブメソッドライブラリ１０５をメモリにロードするネイティブメソッドライブラリロード部１１０の処理である。ステップ２０５は、入力されたバイトコードをＪａｖａＶＭ内で実行する際に必要なメモリを確保するメモリ確保部１１１の処理である。ステップ２０６は、実際にバイトコードを実行するバイトコード実行部１１２の処理である。ステップ２０７及び２０８は、ネイティブメソッドライブラリを実行するネイティブメソッドライブラリ実行部１１３の処理である。ステップ２０９及び２１０は、ネイティブメソッドライブラリを実行したとき、不正なメモリアクセスが発生したかどうかを検出する不正メモリアク

セス検出部１１４の処理である。

【００１３】

バイトコードが入力されたとき、クラスライブラリロード部１０９は、ステップ２０１で他にも必要なバイトコードがあるかどうかを判断する。他にも必要なバイトコードがある場合は、ステップ２０２でバイトコードを収めたクラスライブラリ１０４からクラスライブラリをメモリにロードする。

【００１４】

ネイティブメソッドライブラリロード部１１０は、ステップ２０３で入力されたバイトコードに、Ｊａｖａ言語以外の言語で記述されたネイティブメソッドライブラリを呼び出す処理があるかどうかを判断する。ネイティブメソッドライブラリを呼び出す処理がある場合は、ステップ２０４でネイティブメソッドライブラリ１０５からネイティブメソッドライブラリをメモリにロードする。

【００１５】

メモリ確保部１１１は、ＪａｖａＶＭ１０６が独自に有するメモリ管理機能を利用して、ステップ２０５でＪａｖａＶＭ内でバイトコードを実行する際に必要なメモリを確保する。ＪａｖａＶＭ１０６は、将来不正なメモリアクセスが発生した場合でも検知できるように、確保したメモリ領域を全てメモリ管理テーブルに登録する。またメモリ確保部１１１は、不必要になったメモリを集める処理も行う。

【００１６】

バイトコード実行部１１２は、ステップ２０６で実際にバイトコードを実行する。ネイティブメソッドライブラリ実行部１１３は、ステップ２０７で現在実行しているバイトコードからＪａｖａ言語以外の言語で記述されたネイティブメソッドライブラリを呼び出すか判断する。ネイティブメソッドライブラリを呼び出す場合は、ステップ２０８でネイティブメソッドライブラリを呼び出し、呼び出したネイティブメソッドライブラリに実行の制御を渡す。

【００１７】

不正メモリアクセス検出部１１４は、ステップ２０９でネイティブメソッドライブラリの実行中、または実行後に不正なメモリアクセスが発生したかどうかを

検出する。不正なメモリアクセスが発生した場合は、ステップ２１０で該当するネイティブメソッドライブラリを外部に報告する。不正メモリアクセス検出部１１４は、エラーメッセージを表示装置に表示するか又は指定されたファイルに出力する。以上のように実行部１０８の処理が終了すると、制御は再びバイトコード読み取り部１０７に戻される。

（１）実施例１

図３は、ＯＳがメモリ保護機能をもちかつマルチスレッド制御を行うシステムにおいて、ステップ２０８から２１０の処理を実装する場合の処理のフローチャートである。ネイティブメソッドライブラリ実行部１１３は、ステップ３０１でこれから呼び出すネイティブメソッドライブラリの処理の中で、不正なメモリアクセスが発生しても検知できるように、ステップ２０５の処理で確保したＪａｖａＶＭ内で使用しているメモリ領域に書き込み禁止のプロテクトをかける。ネイティブメソッドライブラリ実行部１１３は、ステップ３０２でネイティブメソッドライブラリを呼び出す。

【００１８】

ネイティブメソッドライブラリ実行中に、プロテクトがかけられたメモリ領域がアクセスされてメモリプロテクション例外が発生したとき、制御はＪａｖａＶＭ１０６に戻される。プロテクトがかけられたメモリ領域をアクセスしたスレッドがＪａｖａＶＭ内で動作しているスレッドの場合には、ＪａｖａＶＭ１０６の例外処理プログラム（不正メモリアクセス検出部１１４）は、ステップ３０３で一旦メモリ領域のプロテクトを外し、ステップ３０４でプロテクトを外したメモリ領域について正常な更新を行う。ステップ３０５で再度メモリ領域にプロテクトをかける処理を行う。ステップ３０３～３０５の処理は、他のスレッドがメモリ領域をアクセスしないようにアトミックに行われる。その後、そのまま元の処理に戻って処理を続行する。またプロテクトがかけられたメモリ領域をアクセスしたスレッドがネイティブメソッドライブラリのスレッドの場合は、ステップ３０６でネイティブメソッドライブラリのプログラムをエラーメッセージとして報告して処理を終了する。

【００１９】

例外が発生せずにネイティブメソッドライブラリの実行が終了し、ＪａｖａＶＭ１０６に制御が戻ってきたとき、ネイティブメソッドライブラリ実行部１１３は、ステップ３０７でメモリにかけたプロテクトを外す。

【００２０】

もしＪａｖａVM内で使用しているメモリ領域にプロテクトをかけない指示があった場合は、ステップ３０１、３０３、３０５、３０７の処理を実行しない。すなわちネイティブメソッドライブラリが不正なメモリアクセスをしないことが確認されたとき、オーバヘッドとなる処理を除去することができる。

【００２１】

図４は、ＯＳがメモリ保護機能をもちかつマルチスレッド制御を行うシステムで、ステップ３０２で呼び出したネイティブメソッドライブラリの実行中に、不正なメモリアクセスが発生したときの例を示す図である。

【００２２】

図４は、不正なメモリアクセスが発生しないＪａｖａVM１０６のシステムで動作するＪａｖａプログラムから、自由にメモリアクセスを行えるシステムで動作するＣ言語で記述されたネイティブメソッドライブラリ４０２の処理が呼ばれた状態を示している。ネイティブメソッドライブラリ（ｆｕｎｃＡ）４０２は、ポインタｉｐのポイント先である領域４０３を更新するつもりが、誤ってライトプロテクトがかけられたメモリ領域４０４内の領域４０５を更新しようとした。この場合、更新しようとしたスレッドがネイティブメソッドライブラリで動作しているスレッドのためメモリプロテクション例外が発生し、ＪａｖａVM１０６の例外処理プログラムは、その時に実行していたネイティブメソッドライブラリ（ｆｕｎｃＡ）を報告して終了する。もしライトプロテクトがかけられたメモリ領域４０４内の領域４０６を更新しようとしたスレッドがＪａｖａVMで動作しているスレッドの場合は、ＪａｖａVM１０６は、そのプロテクトを外して領域４０６の更新を許し、再びメモリ領域４０４にライトプロテクトをかける。

（２）実施例２

図５は、ＯＳがメモリ保護機能をもたずかつマルチスレッド制御を行うシステムにおいて、ステップ２０８から２１０の処理を実装する場合の処理のフローチ

ャートである。ネイティブメソッドライブラリ実行部１１３は、ステップ５０１でステップ２０５の処理で確保したＪａｖａＶM内で使用しているメモリ領域についてその内容のチェックサムを求めて何かの記憶領域に退避する。この処理は、他のスレッドがメモリ領域を更新してチェックサムも更新しないようにアトミックに行われる。

【００２３】

複数のスレッドがネイティブメソッドライブラリを実行している時に、不正なメモリアクセスが発生した場合、どのスレッドで実行しているネイティブメソッドライブラリに問題があるのか特定できなくなる。この状態を避けるためにネイティブメソッドライブラリ実行部１１３は、ステップ５０２で他のＪａｖａＶＭのスレッドがネイティブメソッドライブラリ１０５を実行している場合は、そのスレッドが実行しているネイティブメソッドライブラリ１０５の実行が終わるまで待つ。その後ネイティブメソッドライブラリ実行部１１３は、ステップ５０３でネイティブメソッドライブラリを呼び出す。

【００２４】

ＪａｖａＶM内で動作しているスレッドがメモリを更新する場合は、ステップ５０４で本来のメモリ更新を許し、ステップ５０５でＪａｖａＶMのスレッドが更新した部分のみの更新前後の差分を計算し、新しいチェックサムによってステップ５０１で退避したチェックサムを更新する。この処理は、他のスレッドがメモリ領域を更新してチェックサムも更新しないようにアトミックに行われる。

【００２５】

ＪａｖａＶM内で動作しているスレッドがメモリを更新する時に、他のＪａｖａＶMのスレッドがネイティブメソッドライブラリを呼び出していない場合は、ステップ５０５の処理を行う必要はない。ネイティブメソッドライブラリのスレッドがメモリ領域を更新する場合は、不正なメモリアクセスがあってもそのまま処理を続行する。

【００２６】

ネイティブメソッドライブラリから処理が戻ってきたとき、ネイティブメソッドライブラリ実行部１１３は、ステップ５０６でステップ２０５の処理で確保し

たＪａｖａＶＭ内で使用しているメモリ領域の内容について現在のチェックサムを求める処理をアトミックに行う。不正メモリアクセス検出部１１４は、ステップ５０７で退避しておいたチェックサムとステップ５０６で求めたチェックサムとを比較する。両者が不一致の場合は、ステップ５０８で直前に呼び出したネイティブメソッドライブラリをエラーメッセージとして外部に報告して処理を終了する。

【００２７】

もしＪａｖａＶＭ内で使用しているメモリ領域のチェックサムを求めない指示があった場合は、ステップ５０１、５０２、５０５から５０８の処理は実行しない。

【００２８】

図６は、ＯＳがメモリ保護機能をもたずかつマルチスレッド制御を行うシステムで、ステップ５０３で呼び出したネイティブメソッドライブラリの実行中に、不正なメモリアクセスが発生したときの例を示す図である。

【００２９】

図６は、不正なメモリアクセスが発生しないＪａｖａＶＭ１０６のシステムで動作するＪａｖａプログラムから、自由にメモリアクセスを行えるシステムで動作するＣ言語で記述されたネイティブメソッドライブラリ４０２の処理が呼ばれた状態を示している。ネイティブメソッドライブラリ実行部１１３は、ネイティブメソッドライブラリ４０２を呼ぶ前にＪａｖａＶＭ内で使用しているメモリ領域４０４内の領域６０６にチェックサムを退避する（ステップ５０１）。ネイティブメソッドライブラリ（ｆｕｎｃＡ）４０２は、ポインタｉｐのポイント先である領域４０３を更新するつもりが、誤ってＪａｖａＶＭ内で使用しているメモリ領域４０４内の領域４０５を更新して、たまたま処理が正常に終了した。この場合、制御は再びネイティブメソッドライブラリ４０２の処理を呼び出したＪａｖａプログラムの呼び出し元に戻ってくる。その直後、ＪａｖａＶＭ内で使用しているメモリ領域４０４のチェックサムを求める処理（ステップ５０６）を実行し、求めたチェックサムと領域６０６に退避しておいたチェックサムの比較を行う（ステップ５０７）。ＪａｖａＶＭ内で使用しているメモリ領域４０４内の領

域４０５が不正に更新されているため比較結果が不一致となり、不正メモリアクセス検出部１１４は、直前に呼ばれたネイティブメソッドライブラリ４０２に問題があることを報告して終了する。もしＪａｖａＶＭで動作しているスレッドがチェックサムによってライトプロテクトがかけられたメモリ領域４０４の領域４０６を更新した場合は、領域４０６の更新前後の差分を求め、領域６０６に退避してあるチェックサムの値を更新する（ステップ５０５）。

【００３０】

なおチェックサムの値を格納する領域６０６は、メモリ領域４０４内に限られず、任意のメモリ又は記憶装置でよい。

（３）実施例３

図７は、ＯＳがメモリ保護機能をもちかつマルチスレッド制御を行うシステムにおいて、ＪａｖａＶＭからネイティブメソッドライブラリを呼び出す時に、ＪａｖａＶＭ内で動作している他のスレッドを停止させることができる場合のステップ２０８～２１０を実装する処理のフローチャートである。実行部１０８は、ステップ７０１でこれからライトプロテクトをかけるＪａｖａＶＭ内で使用しているメモリ領域をＪａｖａＶＭで起動している他のスレッドがアクセスしないように、現在起動しているＪａｖａＶＭ内の他のスレッドの実行を停止させる。実行部１０８は、ステップ７０２でこれから呼び出すネイティブメソッドライブラリの処理の中で、不正なメモリアクセスが発生しても検知できるように、ステップ２０５の処理で確保したＪａｖａＶＭ内で使用しているメモリ領域にプロテクトをかける。

【００３１】

ネイティブメソッドライブラリ実行部１１３は、ステップ７０３でネイティブメソッドライブラリを呼び出す。ネイティブメソッドライブラリ実行中に、ライトプロテクトがかけられたメモリ領域がアクセスされてメモリプロテクション例外が発生した場合、ステップ７０４で、止めたＪａｖａＶＭ内の他のスレッドを再開させる。メモリ領域をアクセスしたスレッドはＪａｖａＶＭのスレッドではないので、不正メモリアクセス検出部１１４は、ステップ７０５でその時に実行していたネイティブメソッドライブラリのプログラムを報告して処理を終了させ

る。

【００３２】

正常にネイティブメソッドライブラリから処理が戻ってきたとき、実行部１０８は、ステップ７０６でメモリにかけたプロテクトを外し、ステップ７０７で、止めていたＪａｖａＶＭ内の他のスレッドを再開させる。

（４）実施例４

図８は、ＯＳがメモリ保護機能をもたずかつマルチスレッド制御を行うシステムにおいて、ＪａｖａＶＭからネイティブメソッドライブラリを呼び出す時に、ＪａｖａＶＭ内で動作している他のスレッドを停止させることができる場合のステップ２０８～２１０を実装する処理のフローチャートである。実行部１０８は、ステップ８０１でこれからチェックサムを求めるＪａｖａＶＭ内で使用しているメモリ領域をＪａｖａＶＭ内で起動している他のスレッドがアクセスしないように、現在起動しているＪａｖａＶＭ内の他のスレッドを全て止める。ネイティブメソッドライブラリ実行部１１３は、ステップ８０２でステップ２０５の処理で確保したＪａｖａＶＭ内で使用しているメモリ領域のチェックサムを求めて何らかの記憶領域に退避する。

【００３３】

ネイティブメソッドライブラリ実行部１１３は、ステップ８０３でネイティブメソッドライブラリを呼び出す。ネイティブメソッドライブラリのスレッドがメモリ領域を更新する場合は、不正なメモリ領域であってもそのまま処理は続行する。

【００３４】

ネイティブメソッドライブラリから処理が戻ってきたとき、ネイティブメソッドライブラリ実行部１１３は、ステップ８０４で確保したＪａｖａＶＭ内で使用しているメモリ領域の現在のチェックサムを求める。次に実行部１０８は、ステップ８０５で、止めていたＪａｖａＶＭ内の他のスレッドを再開させる。

【００３５】

不正メモリアクセス検出部１１４は、ステップ８０６で退避しておいたチェックサムとステップ８０４で求めたチェックサムとを比較する。両者が不一致の場

合は、不正メモリアクセス検出部１１４は、ステップ８０７で直前に呼び出したネイティブメソッドライブラリをエラーメッセージとして外部に報告して処理を終了する。

【００３６】

なお上記実施例２及び実施例４では、チェックサムを計算したが、チェックサムを計算する代わりにハッシュ関数を用いたりデータ圧縮した結果を用いるなど、上記メモリ領域の内容を入力とする関数手続きによって得られる結果のコード情報であって上記メモリ領域の内容に一意又は高い確率で対応するコード情報が得られるような関数手続きであれば何でもよい。もちろん上記メモリ領域の内容をそのままメモリなどの記憶装置に退避する場合も含まれる。

【００３７】

以上の実施例において、必要とするＪａｖａＶＭ内のスレッド管理、アトミック処理等は、従来からＪａｖａＶＭに備わっている機能であるため詳述しない。

【００３８】

【発明の効果】

本発明によれば、不正なメモリアクセスが発生しないシステムで動作しているプログラムから呼ばれた自由にメモリアクセスを行えるシステムで動作するプログラムが実行中又は実行後早期に不正なメモリアクセスが発生したことを検出できる。

【図面の簡単な説明】

【図１】

実施形態のＪａｖａＶＭの構成図である。

【図２】

実施形態の実行部１０８の処理手順を示すフローチャートである。

【図３】

実施例１の処理手順を示すフローチャートである。

【図４】

実施例１における不正メモリアクセスを説明する図である。

【図５】

実施例２の処理手順を示すフローチャートである。

【図６】

実施例２における不正メモリアクセスを説明する図である。

【図７】

実施例３の処理手順を示すフローチャートである。

【図８】

実施例４の処理手順を示すフローチャートである。

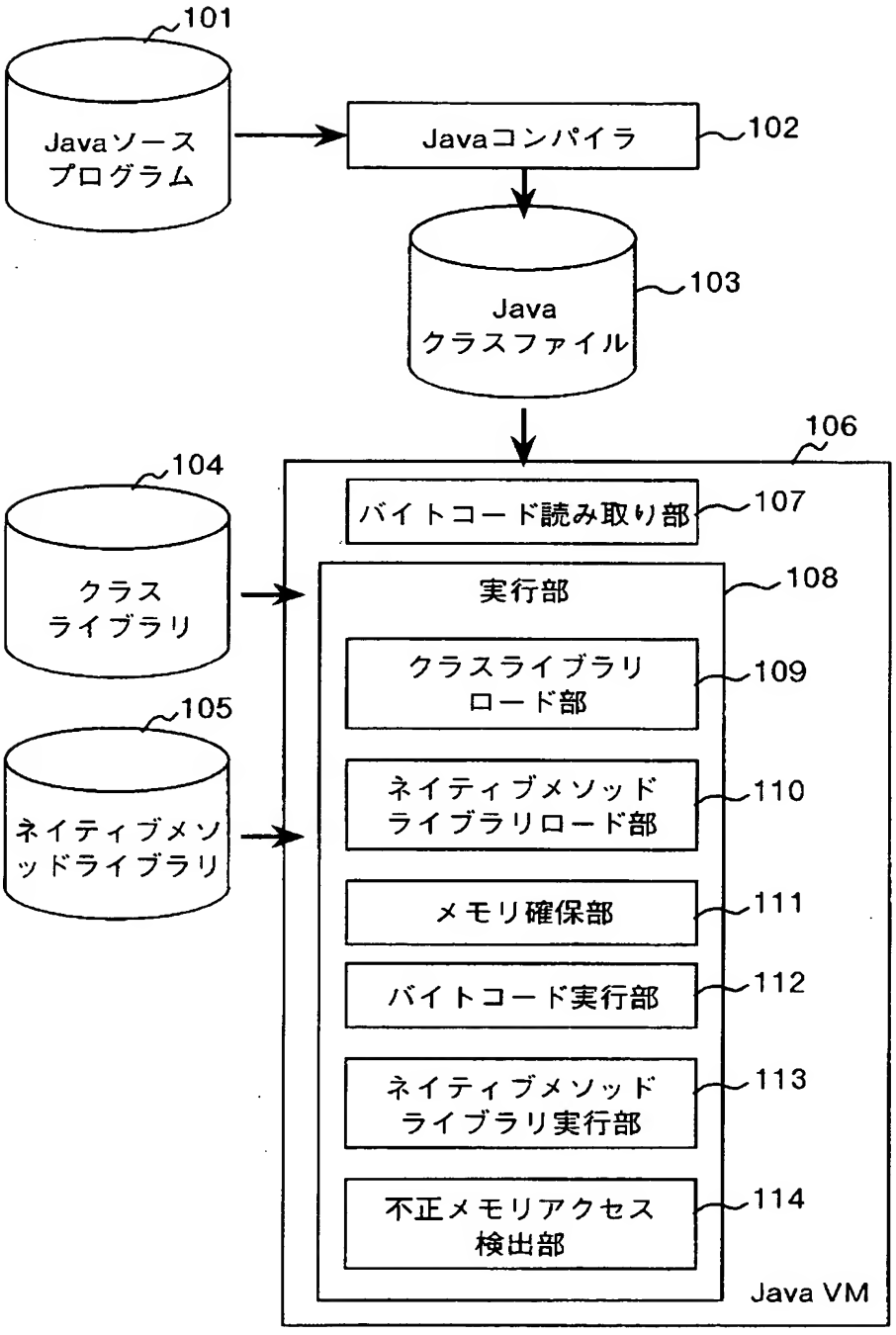
【符号の説明】

１０３…Ｊａｖａクラスファイル、１０５…ネイティブメソッドライブラリ、１０６…ＪａｖａＶＭ、１１３…ネイティブメソッドライブラリ実行部、１１４…不正メモリアクセス検知部。

【書類名】　図面

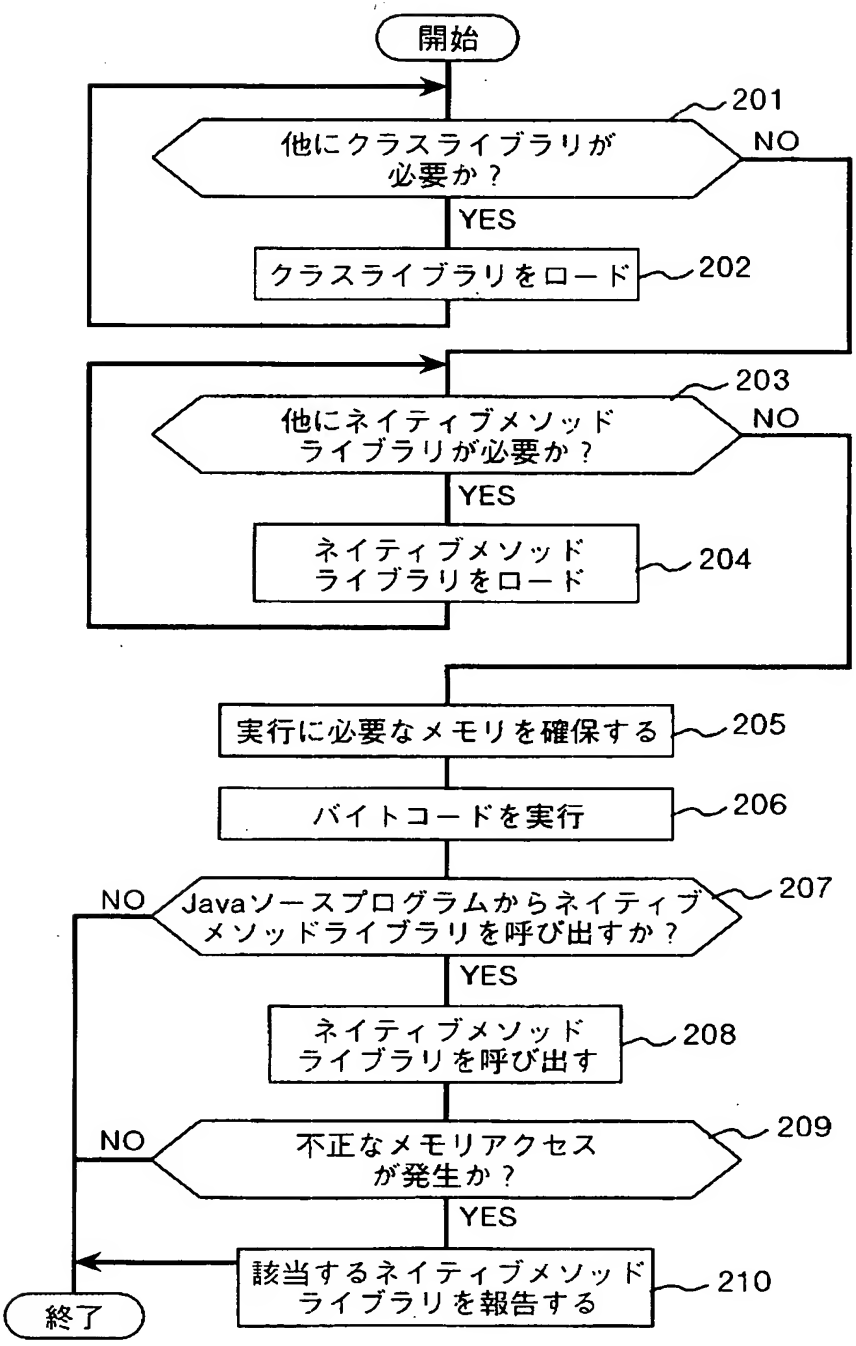
【図１】

図　1


101
Javaソース
プログラム
Javaコンパイラ
102
Java
クラスファイル
103
106
104
クラス
ライブラリ
バイトコード読み取り部
107
実行部
108
クラスライブラリ
ロード部
109
105
ネイティブメソ
ッドライブラリ
ネイティブメソッド
ライブラリロード部
110
メモリ確保部
111
バイトコード実行部
112
ネイティブメソッド
ライブラリ実行部
113
不正メモリアクセス
検出部
114
Java VM

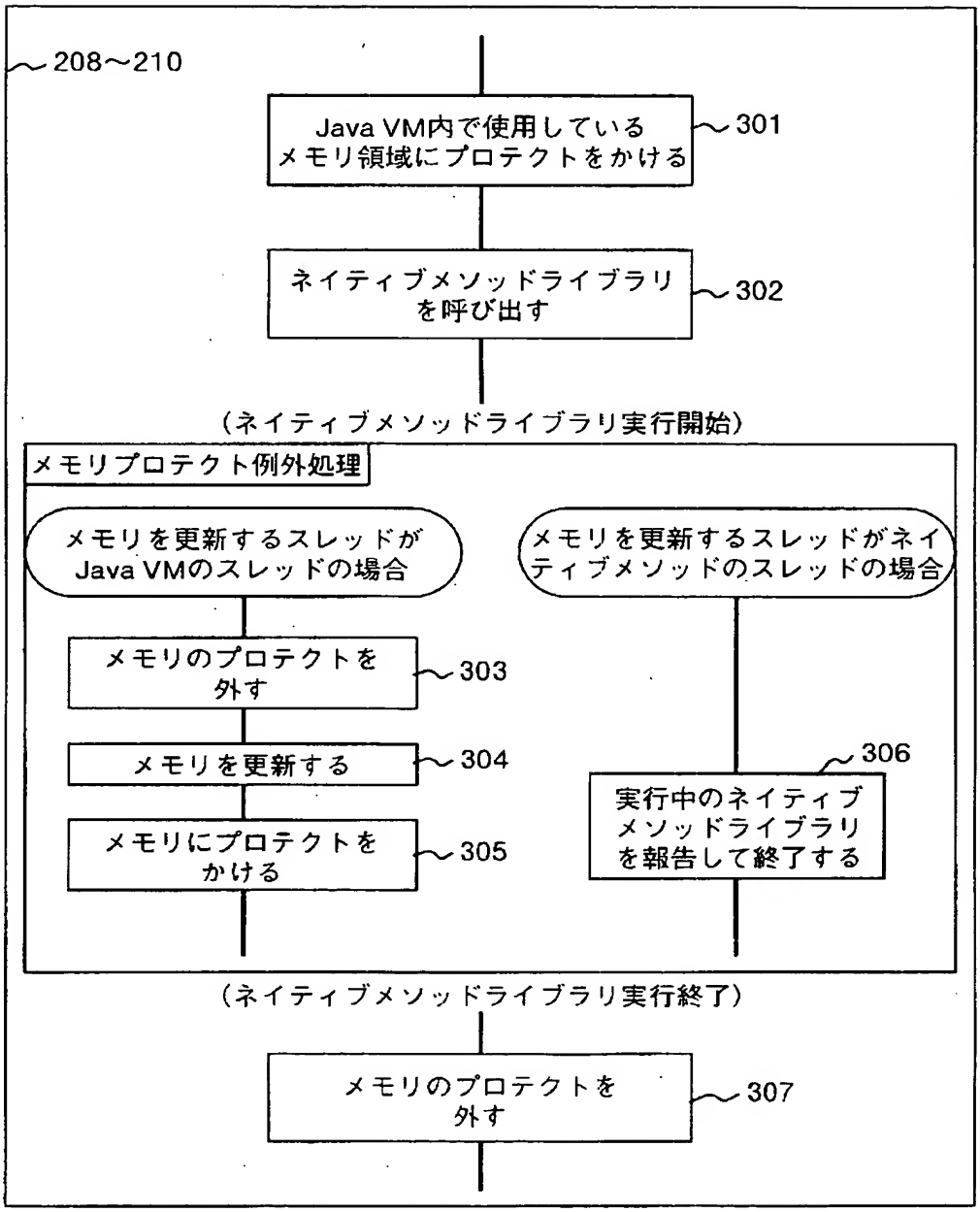
【図２】

図　2


開始
201
他にクラスライブラリが必要か？
NO
YES
クラスライブラリをロード
202
203
他にネイティブメソッドライブラリが必要か？
NO
YES
ネイティブメソッドライブラリをロード
204
実行に必要なメモリを確保する
205
バイトコードを実行
206
NO
Javaソースプログラムからネイティブメソッドライブラリを呼び出すか？
207
YES
ネイティブメソッドライブラリを呼び出す
208
NO
不正なメモリアクセスが発生か？
209
YES
該当するネイティブメソッドライブラリを報告する
210
終了

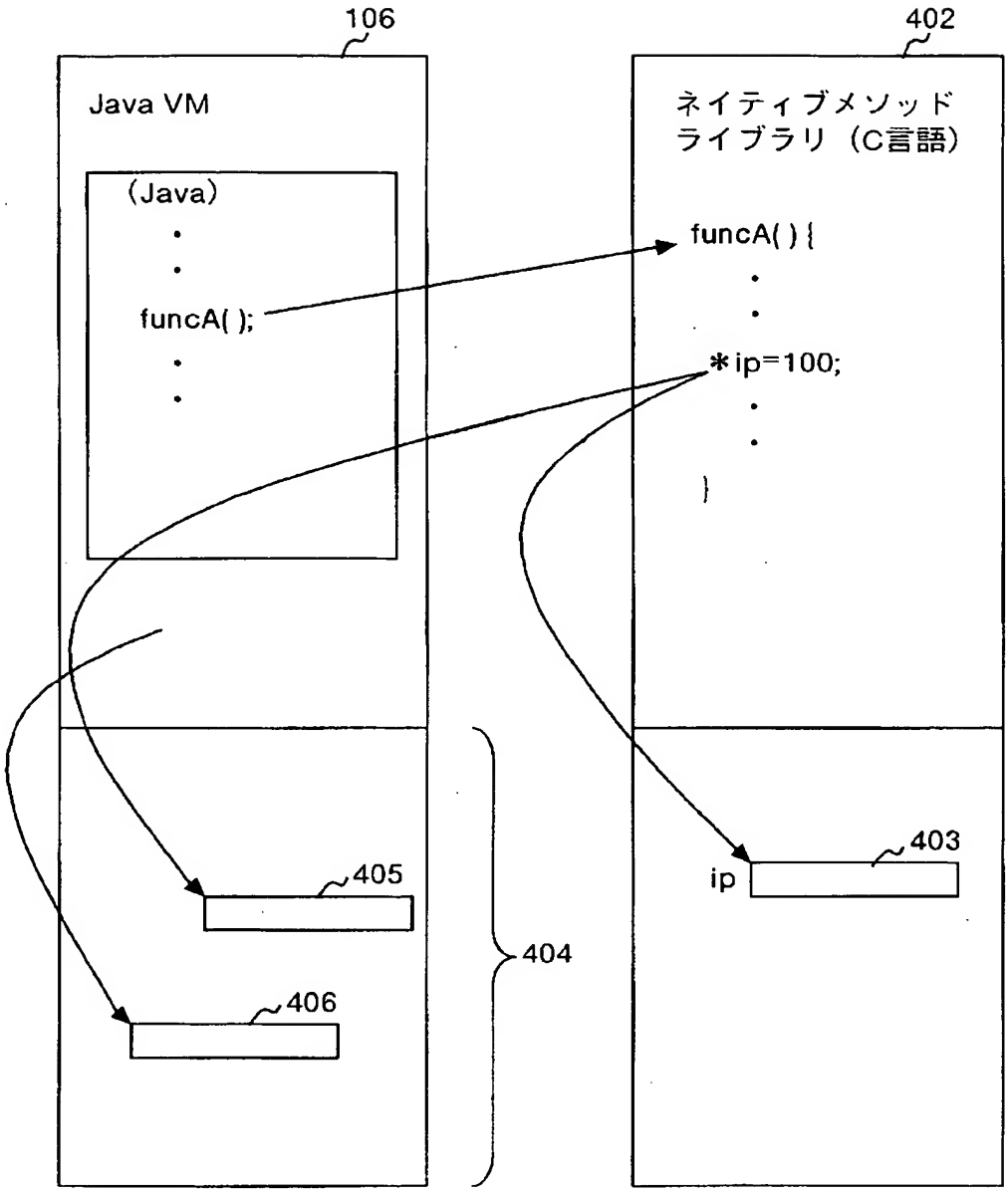
【図３】

図　3


208～210
Java VM内で使用している
メモリ領域にプロテクトをかける
301
ネイティブメソッドライブラリ
を呼び出す
302
（ネイティブメソッドライブラリ実行開始）
メモリプロテクト例外処理
メモリを更新するスレッドが
Java VMのスレッドの場合
メモリを更新するスレッドがネイ
ティブメソッドのスレッドの場合
メモリのプロテクトを
外す
303
メモリを更新する
304
メモリにプロテクトを
かける
305
306
実行中のネイティブ
メソッドライブラリ
を報告して終了する
（ネイティブメソッドライブラリ実行終了）
メモリのプロテクトを
外す
307

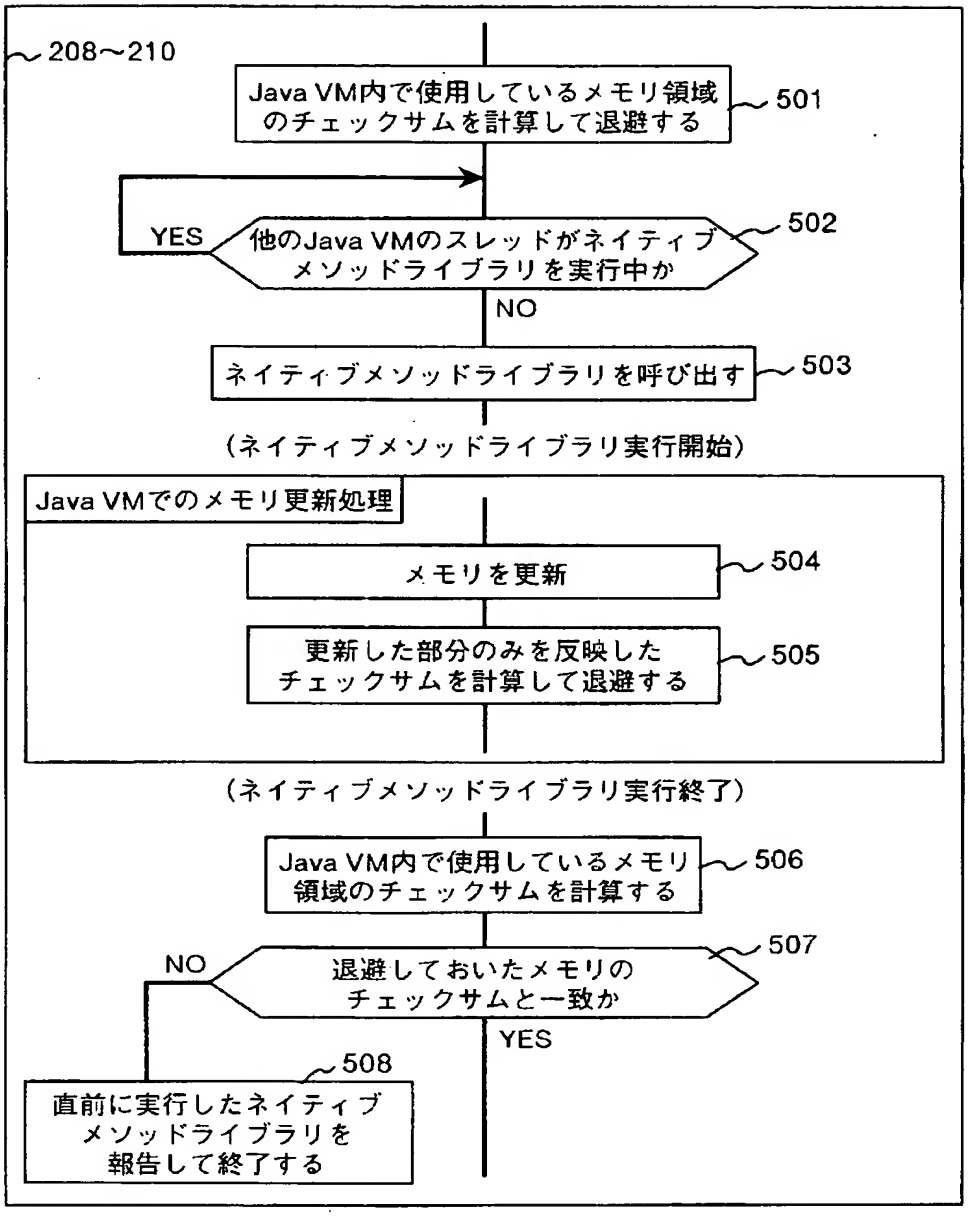
【図４】

図　4


106
402
Java VM
ネイティブメソッド
ライブラリ（C言語）
(Java)
funcA( );
funcA( ) {
*ip=100;
}
405
406
404
403
ip

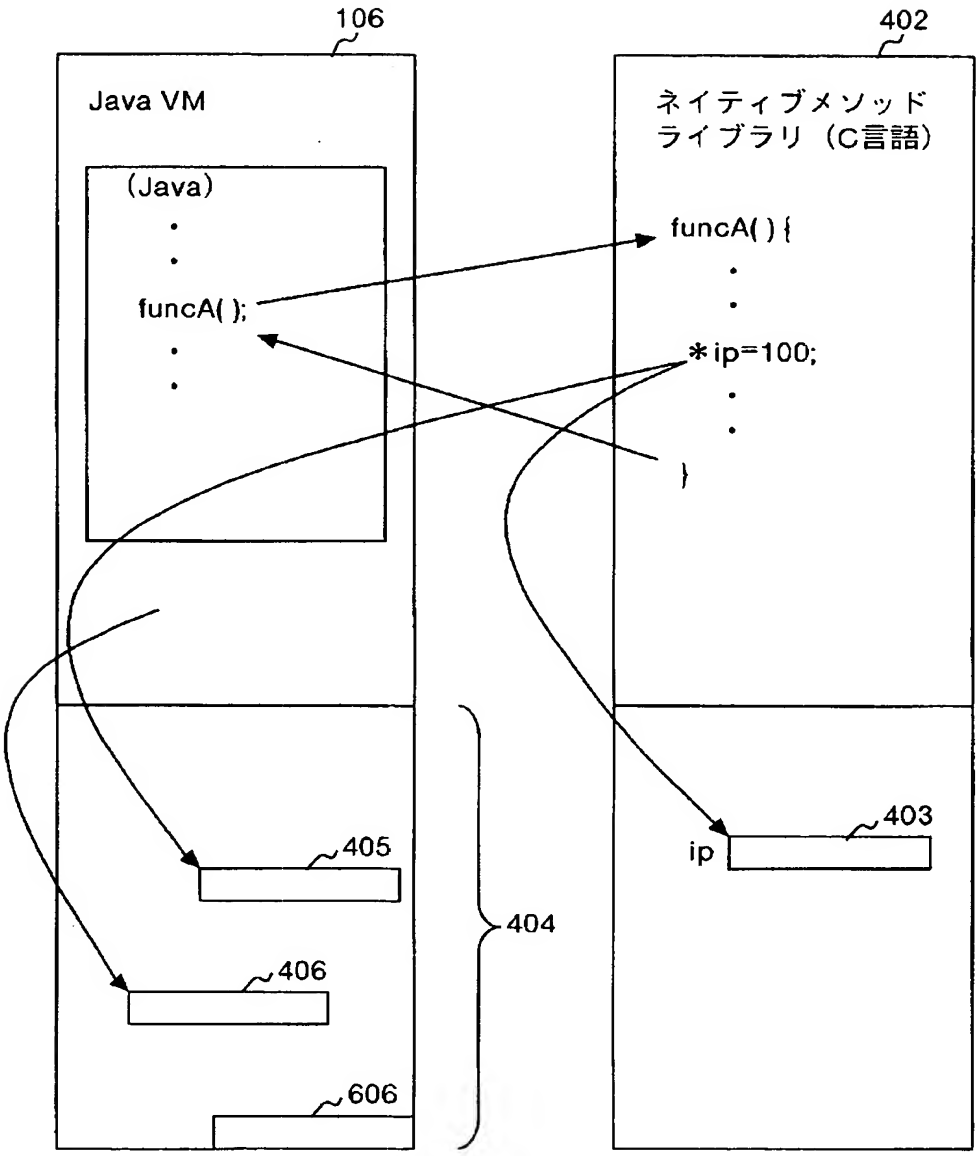
【図５】

図　5


208～210
Java VM内で使用しているメモリ領域のチェックサムを計算して退避する
501
YES
他のJava VMのスレッドがネイティブメソッドライブラリを実行中か
502
NO
ネイティブメソッドライブラリを呼び出す
503
（ネイティブメソッドライブラリ実行開始）
Java VMでのメモリ更新処理
メモリを更新
504
更新した部分のみを反映したチェックサムを計算して退避する
505
（ネイティブメソッドライブラリ実行終了）
Java VM内で使用しているメモリ領域のチェックサムを計算する
506
NO
退避しておいたメモリのチェックサムと一致か
507
YES
508
直前に実行したネイティブメソッドライブラリを報告して終了する

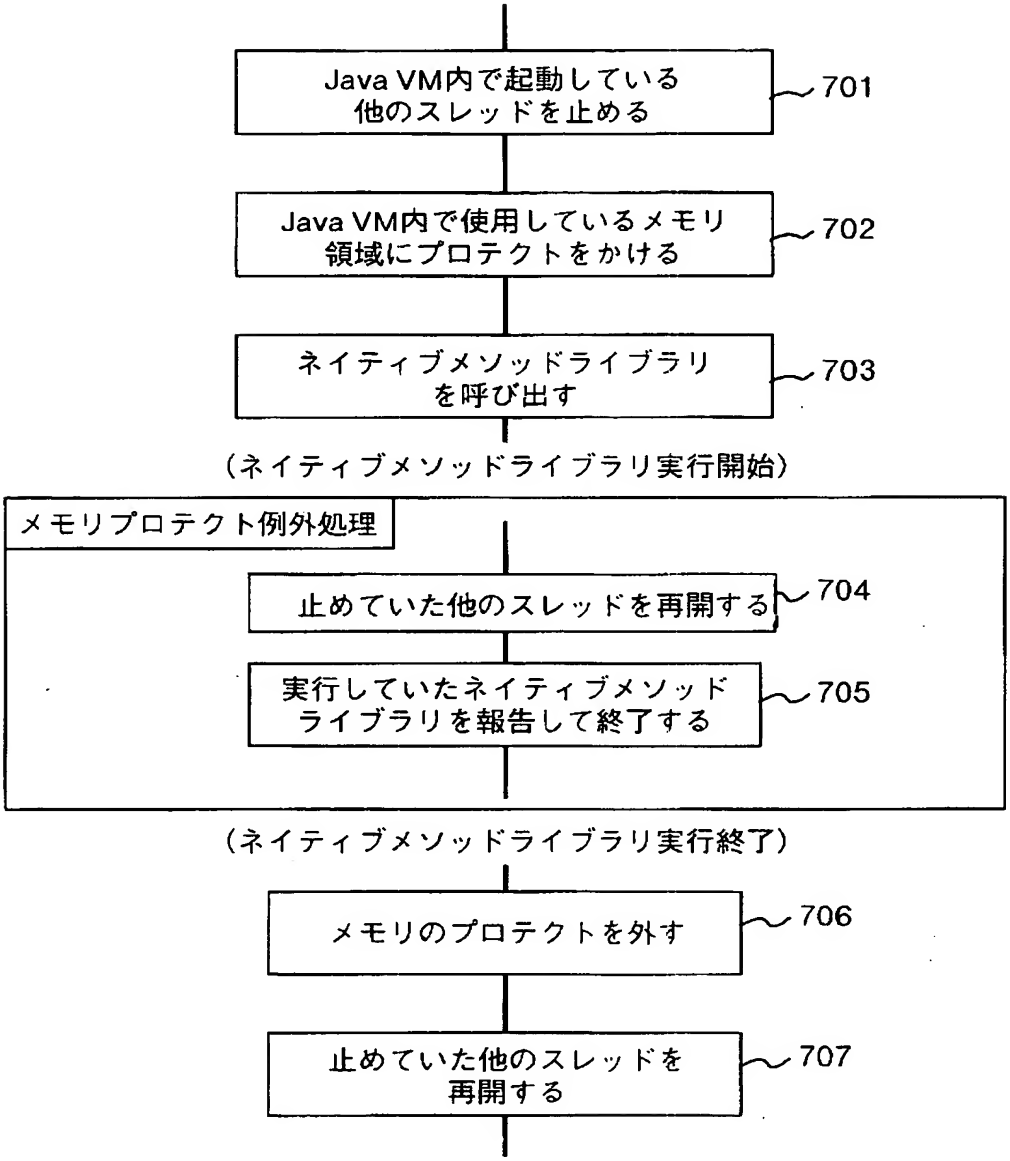
【図６】

図　6


106
402
Java VM
ネイティブメソッド
ライブラリ（C言語）
(Java)
funcA( );
funcA( ) {
*ip=100;
}
405
406
606
404
403
ip

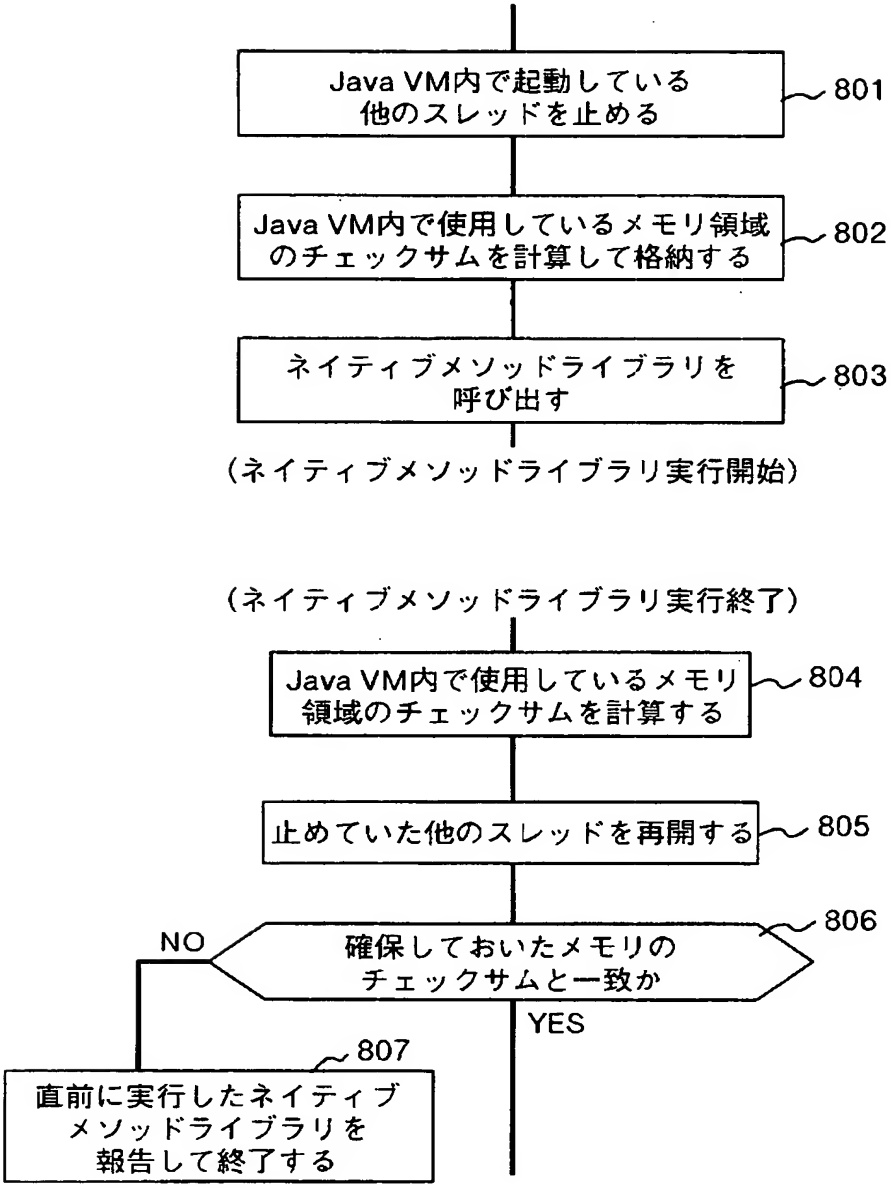
【図７】

図　7


Java VM内で起動している
他のスレッドを止める
701
Java VM内で使用しているメモリ
領域にプロテクトをかける
702
ネイティブメソッドライブラリ
を呼び出す
703
（ネイティブメソッドライブラリ実行開始）
メモリプロテクト例外処理
止めていた他のスレッドを再開する
704
実行していたネイティブメソッド
ライブラリを報告して終了する
705
（ネイティブメソッドライブラリ実行終了）
メモリのプロテクトを外す
706
止めていた他のスレッドを
再開する
707

【図８】

図　8


Java VM内で起動している
他のスレッドを止める
801
Java VM内で使用しているメモリ領域
のチェックサムを計算して格納する
802
ネイティブメソッドライブラリを
呼び出す
803
（ネイティブメソッドライブラリ実行開始）
（ネイティブメソッドライブラリ実行終了）
Java VM内で使用しているメモリ
領域のチェックサムを計算する
804
止めていた他のスレッドを再開する
805
806
NO
確保しておいたメモリの
チェックサムと一致か
YES
807
直前に実行したネイティブ
メソッドライブラリを
報告して終了する

【書類名】　要約書

【要約】

【課題】　不正なメモリアクセスが発生しないシステムで動作するプログラムから呼ばれた自由にメモリアクセスを行えるシステムで動作するプログラムが不正なメモリアクセスをしたことを早期に検出する。

【解決手段】　ＪａｖａＶＭ１０６の実行部１０８は、読み込まれたＪａｖａのバイトコードを実行する。ネイティブメソッドライブラリ実行部１１３は、ネイティブメソッドライブラリを呼び出して実行させる。不正メモリアクセス検出部１１４は、ネイティブメソッドライブラリの実行中又は実行後に、メモリ確保部１１１が確保したメモリ領域についてネイティブメソッドライブラリによる不正なメモリアクセスを検出する。

【選択図】図１

特願２００３－１１８６０２

出　願　人　履　歴　情　報

識別番号　［０００００５１０８］

１．変更年月日　１９９０年　８月３１日
［変更理由］　新規登録
住　所　東京都千代田区神田駿河台４丁目６番地
氏　名　株式会社日立製作所